ELSEVIER

Available online at www.sciencedirect.com

SCIENCE DIRECT®

Discrete Applied Mathematics 146 (2005) 287–300

DISCRETE APPLIED MATHEMATICS

www.elsevier.com/locate/dam

Minimization of ordered, symmetric half-products

Wieslaw Kubiak

Faculty of Business Administration, Memorial University of Newfoundland, St. John's, NL, Canada A1B 3X5

Received 14 December 2000; received in revised form 28 January 2004; accepted 29 July 2004
Available online 18 December 2004

Abstract

We introduce a class of pseudo-Boolean functions called ordered, symmetric half-products. The class includes a number of well known scheduling problems. We study sets of dominating solutions for minimization of the half-products, and we show their fully polynomial time approximation schemes that use a natural rounding scheme to obtain ε-solutions in $O(n^2/\varepsilon)$ time.

Keywords: Pseudo-Boolean functions; Scheduling; Fully polynomial time approximation schemes; Half-products; Optimization

1. Introduction

A half-product is a pseudo-Boolean function of the form

$$H(x) = H(x_1, \ldots, x_n) = \sum_{1 \leqslant j < i \leqslant n} a_i b_j x_i x_j - \sum_{1 \leqslant i \leqslant n} c_i x_i,$$

where $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$ are vectors of non-negative integers. The notion of half-products was introduced by Badics and Boros [3], and independently by Kubiak [18]. It has attracted attention since a number of scheduling problems can be formulated as half-product minimization problems. These include scheduling two machines to minimize total weighted completion time (WCT), Jurisch et al. [14], scheduling two machines to minimize makespan (MAKS), Jurisch et al. [14], scheduling a single machine to minimize completion time variance (CTV), Badics and Boros [3], and Kubiak [18] or to minimize

E-mail address: wkubiak@mun.ca (W. Kubiak).
URL: http://wawel.busi.mun.ca

0166-218X/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2004.07.007

agreeably weighted completion variance (AWCTV), Cheng and Kubiak [7], scheduling a single machine to minimize total weighted earliness and tardiness (WET), Jurisch et al. [14], and scheduling with controllable processing times (CONT), Janiak et al. [13]. The definitions of all these problems as well as their half-products are given in the Appendix. All these scheduling problems have been proven to be NP-hard in the ordinary sense, and fully polynomial time approximation schemes have been developed for all of them, see the Appendix for details.

It is worth observing that half-products occur also in some models of physics, see for instance the energy in the infinite range Mattis model of a spin-glass, Mattis [21], and also Amit [1].

Hammer et al. [10] studied exact optimization of half-products.

A subclass of half-products, referred to as *symmetric* half-products, of the form

$$F_{a,b}(x) = F_{a,b}(x_1, \ldots, x_n) = -\sum_{1 \leqslant j < i \leqslant n} a_i b_j x_i \oplus x_j,$$

where $x_i \oplus x_j = \overline{x}_i x_j + x_i \overline{x}_j = x_i + x_j - 2x_i x_j$ and $\overline{x}_i = 1 - x_i$, was introduced by Kubiak [18]. We follow here the standard notation for Boolean functions, see for instance Wegener [27], where the sign $\oplus$ denotes the exclusive OR also called the parity function. Let $e = (1, \ldots, 1)$ be an n-dimensional unit vector. We have the following two relations for the symmetric half-products exploited later in the paper.

$$F_{a,b}(x) = F_{a-e,b}(x) + F_{e,b}(x), \tag{1}$$

for a *positive* vector a and

$$F_{a,b}(x) = F_{a,b-e}(x) + F_{a,e}(x), \tag{2}$$

for a *positive* vector b. For convenience, we shall drop the subscripts a and b from the half-product notation whenever this causes no confusion.

Symmetric half-products, just like all half-products, have all quadratic terms with non-negative coefficients, and thus, Nemhauser et al. [23], are supermodular functions. The minimization of supermodular functions is known in general to be NP-hard in the strong sense, see for instance Nemhauser and Wolsey [22], hence the supermodularity of half-products, in itself, does not seem to help much in finding efficient optimization and approximation algorithms for the half-products.

In this paper, we focus in particular on symmetric half-products with at least one of the two vectors a or b being either *ascending* or *descending*. A vector c is *ascending* if $c_1 \leqslant c_2 \leqslant \cdots \leqslant c_n$, similarly, a vector c is *descending* if $c_1 \geqslant c_2 \geqslant \cdots \geqslant c_n$. We refer to these half-products as *ordered*, symmetric half-products. It is worth observing that if the half-product on the left hand side in relations (1) and (2) is ordered, then so are both half-products on the right hand side of (1) and (2), furthermore, the order is the same on both sides. Though the minimization of ordered, symmetric half-products still remains generally NP-hard in the ordinary sense (both the CTV and the MAKS half-products are symmetric and ordered, see the Appendix) efficient fully polynomial time approximation scheme of Badics and Boros [3] applies to them as a special class of

half-products. Their scheme runs in $O((n^2 \ln A)/\varepsilon)$ time, where $A = \sum_{1 \leqslant i \leqslant n} a_i$ and works for any half-product. We show in this paper that a more efficient fully polynomial time approximation scheme based on a *natural* rounding scheme exist for ordered, symmetric half-products whenever b is descending *or* a is ascending. The schemes run in $O(n^2/\varepsilon)$ time.

On the other hand if b is ascending *and* a descending, we show that an optimal solution is alternating, and thus its value can be calculated in linear time. Finally, we show dominating and optimal solutions for some other special ordered, symmetric half-products.

We close this short introduction remarking that half-products are special cases of well-known pseudo-Boolean quadratic functions, see Boros and Hammer [4], Hansen et al. [11], for which the general minimization problem does *not* admit *polynomial approximation scheme* if $P \neq NP$, see Arora et al. [2].

2. Optimization of half-products and their approximation schemes

A half-product optimization problem is an optimization problem with the objective to find a vector $x^* \in \{0, 1\}^n$ that minimizes a function of the form

$$S(x) = D + H(x),$$

where $H(x)$ is a half-product and D is a non-negative constant independent of x. Problems MAKS, CTV, AWCTV, WET, WCT, and CONT are all half-product optimization problems, also, all of them are NP-hard in the ordinary sense. Furthermore, the CTV problem is an example of a symmetric half-product and WET is an example of half-product which is not symmetric. The CTV and AWCTV are also examples of ordered half-products, see the Appendix for details.

Badics and Boros [3] propose a fully polynomial time approximation scheme for half-product minimization. Their scheme finds an ε-solution to any half-product $H(x)$ in $O((n^2 \ln A)/\varepsilon)$ time. This scheme can be used to find ε-solutions to the half-product optimization problems at the cost of additional computational time which is due to the fact that the values $|S(x)|$ may be smaller than the values $|H(x)|$. The ratio, we call it an approximation ratio or an a-ratio for S

$$|H(x^*)/S(x^*)|,$$

where x^* is an optimal solution to $S(x)$, is a good measure of this additional time as shown by the following simple observation.

Lemma 1. *Let x^* and x^0 be an optimal and an ε-solution, respectively, to the problem of minimizing $H(x)$. If $|H(x^*)/S(x^*)| \leqslant f(n)$ for some positive function f of n, then x^0 is an $f(n) \cdot \varepsilon$-solution to the problem of minimizing $S(x)$.*

Proof. We have

$$S(x^0) - S(x^*) = H(x^0) - H(x^*) \leqslant \varepsilon |H(x^*)| \leqslant \varepsilon \cdot f(n) \cdot |S(x^*)|. \qquad \square$$

For some problems, $f(n)$ is constant. Consider for instance WCT. Jurisch et al. [14] observed that

$$WCT(x) = \sum_{1 \leqslant j \leqslant i \leqslant n} w_i p_j - F(x),$$

where

$$F(x) = \sum_{1 \leqslant j < i \leqslant n} w_i p_j x_i \oplus x_j.$$

It follows from Eastman et al. [8] that

$$WCT(x^*) \geqslant 1/2 \sum_{1 \leqslant j < i \leqslant n} w_i p_j. \tag{3}$$

Consequently,

$$F(x^*) \leqslant \sum_{1 \leqslant j < i \leqslant n} w_i p_j \leqslant 2WCT(x^*),$$

which proves that an ε-solution for $F(x)$ is a 2ε-solution for $WCT(x)$, and consequently $f(n) = 2$. Similar results can be shown for MAKS, where $f(n) = 1$, and CTV, where $f(n) = 3$, see Kubiak et al. [19]. Cheng and Kubiak [7] show that $f(n) = 4n^2 - 1$ for AWCTV. However, for WET, $f(n)$ cannot be bounded by any polynomial of n which we show below:

Jurisch et al. [14] observed that

$$WET(x) = \sum_{1 \leqslant j \leqslant i \leqslant n} w_i p_j - G(x),$$

where

$$G(x) = \sum_{1 \leqslant j < i \leqslant n} w_i p_j x_i \oplus x_j + \sum_{1 \leqslant i \leqslant n} w_i p_i x_i.$$

We have

$$G(x^*) \geqslant 1/2 \sum_{1 \leqslant j \leqslant i \leqslant n} w_i p_j.$$

This inequality follows immediately from the following lemma.

Lemma 2. *For any optimal solution x^* to a symmetric half-product $F(x)$,*

$$F(x^*) \geqslant \frac{1}{2} \sum_{1 \leqslant j < i \leqslant n} a_i b_j.$$

Proof. Let us observe that $F(x)$ is a multilinear function since $x_i \oplus x_j = x_i + x_j - 2x_i x_j$. Rosenberg [24] (see also Hansen et al. [11]) observes that the maximum of a multilinear

function over the unit hypercube $[0, 1]^n$ is attained in at least one vertex of that hypercube, i.e. in a point $x^* \in \{0, 1\}^n$. Therefore, we have

$$F(x^*) \geqslant F\left(\frac{1}{2}, \ldots, \frac{1}{2}\right) = \frac{1}{2} \sum_{1 \leqslant j < i \leqslant n} a_i b_j,$$

which ends the proof. □

On the other hand, for $x_1 = \cdots = x_n = 1$

$$WET(x^*) \leqslant \sum_{1 \leqslant j < i \leqslant n} w_i p_j.$$

Consequently, by defining

$$p_i = M(p_1 + \cdots + p_{i-1})$$

for $i = 2, \ldots, n$, and choosing M arbitrarily large, for instance $M = 2^n$, we can make the ratio

$$F(x^*)/WET(x^*) \geqslant M$$

arbitrarily large. Therefore, the fully polynomial time approximation scheme of Badics and Boros [3] does not *imply* a fully polynomial time approximation scheme for WET, since the *a*-ratio for WET is not polynomially bounded. However, such a scheme for WET exists, as was shown by Kovalyov and Kubiak [16].

3. ϵ-solutions

We show fully polynomial time approximation schemes for ordered, symmetric half-products with *descending b*, Theorem 1, or *ascending a*, Theorem 2. The schemes are based on the row-column pair of dynamic programs developed by Kubiak [18] originally for the CTV problem but applicable to any symmetric half-product. The schemes use rounding of *b* vector and *a* vector, respectively, and each works in $O(n^2/\varepsilon)$ time.

Theorem 1. *An ε-solution to the symmetric half-product problem with*

$$b_1 \geqslant b_2 \geqslant \cdots \geqslant b_{n-1} \geqslant 0$$

can be found in $O(n^2/\varepsilon)$ time.

Proof. Let us find y that maximizes

$$F'(x) = \sum_{1 \leqslant j < i \leqslant n} a_i b'_j x_i \oplus x_j,$$

where

$$b'_j = \left\lfloor \frac{b_j}{\delta} \right\rfloor \quad \text{and} \quad \delta = \frac{\varepsilon}{2}\overline{b} = \frac{\varepsilon}{2} \frac{\sum_{1 \leqslant j \leqslant n-1} b_j}{n-1}.$$

(i) y can be found in $O(n\sum_{1\leqslant j\leqslant n-1} b'_j)$ time by a dynamic programming algorithm of Kubiak [18]. This algorithm solves the following recurrence relation:

$$h(k, B) = \max\{h(k+1, B) + a_k B, h(k+1, B+b_k) + a_k(B_{k-1} - B)\} \tag{4}$$

for $h(1, 0)$ with the boundary condition $h(n+1, B) = 0$ for all B. We have $k = 1, \ldots, n$ and $B = 0, \ldots, B_{k-1}$, where $B_{k-1} = \sum_{i=1}^{k-1} b_i$, in (4). The solution can be found in $O(n\sum_{i=1}^{n-1} b_i)$ time for the original vector b, which becomes $O(n^2/\varepsilon)$ time for the rounded vector b'.

(ii) y is an ε-solution since

$$\begin{aligned} F(y) &= \sum_{1\leqslant j<i\leqslant n} a_i b_j y_i \oplus y_j \geqslant \delta \sum_{1\leqslant j<i\leqslant n} a_i \lfloor b_j/\delta \rfloor y_i \oplus y_j \\ &= \delta F'(y) \geqslant \delta F'(x^*) \\ &= \delta \sum_{1\leqslant j<i\leqslant n} a_i \lfloor b_j/\delta \rfloor x_i^* \oplus x_j^* \geqslant \delta \sum_{1\leqslant j<i\leqslant n} a_i (b_j/\delta - 1) x_i^* \oplus x_j^* \\ &= F(x^*) - \delta \sum_{1\leqslant j<i\leqslant n} a_i x_i^* \oplus x_j^*. \end{aligned}$$

Thus

$$\begin{aligned} F(x^*) - F(y) &\leqslant \delta \sum_{1\leqslant j<i\leqslant n} a_i x_i^* \oplus x_j^* = \varepsilon\bar{b}/2 \sum_{1\leqslant j<i\leqslant n} a_i x_i^* \oplus x_j^* \\ &\leqslant \varepsilon\bar{b}/2 \sum_{2\leqslant i\leqslant n} (i-1) a_i. \end{aligned}$$

By Lemma 2

$$F(x^*) \geqslant 1/2 \sum_{1\leqslant j<i\leqslant n} a_i b_j.$$

Therefore, it suffices to prove that

$$\bar{b} \sum_{2\leqslant i\leqslant n} (i-1) a_i \leqslant \sum_{1\leqslant j<i\leqslant n} a_i b_j = \sum_{2\leqslant i\leqslant n} a_i (b_1 + \cdots + b_{i-1}).$$

This, however, holds since

$$b_1 \geqslant b_2 \geqslant \cdots \geqslant b_{n-1} \geqslant 0$$

and consequently

$$\bar{b} \leqslant (b_1 + \cdots + b_{i-1})/(i-1). \qquad \square$$

Theorem 2. *An ε-solution to the symmetric half-product problem with*

$$0 \leqslant a_2 \leqslant a_3 \leqslant \cdots \leqslant a_n$$

can be found in $O(n^2/\varepsilon)$ time.

Proof. Let us find y that maximizes

$$F'(x) = \sum_{1 \leqslant j < i \leqslant n} a'_i b_j x_i \oplus x_j,$$

where

$$a'_i = \left\lfloor \frac{a_i}{\delta} \right\rfloor \quad \text{and} \quad \delta = \frac{\varepsilon}{2}\overline{a} = \frac{\varepsilon}{2}\frac{\sum_{2 \leqslant i \leqslant n} a_i}{n-1}.$$

(i) y can be found in $O(n \sum_{2 \leqslant i \leqslant n} a'_i)$ time by a dynamic programming algorithm of Kubiak [18]. This algorithm solves the following recurrence relation:

$$g(k, A) = \max\{g(k-1, A) + b_k A, g(k-1, A + a_k) + b_k(A_k - A)\} \tag{5}$$

for $g(n, 0)$ with the boundary condition $g(0, A) = 0$ for all A. We have $k = 1, \ldots, n$ and $A = 0, \ldots, A_k$, where $A_k = \sum_{i=k+1}^{n} a_i$, in (5). The solution can be found in $O(n \sum_{i=2}^{n} a_i)$ time for the original vector a, which becomes $O(n^2/\varepsilon)$ time for the rounded vector a'.

(ii) y is an ε-solution since

$$\begin{aligned} F(y) &= \sum_{1 \leqslant j < i \leqslant n} a_i b_j y_i \oplus y_j \geqslant \delta \sum_{1 \leqslant j < i \leqslant n} \lfloor a_i/\delta \rfloor b_j y_i \oplus y_j \\ &= \delta F'(y) \geqslant \delta F'(x^*) \\ &= \delta \sum_{1 \leqslant j < i \leqslant n} \lfloor a_i/\delta \rfloor b_j x_i^* \oplus x_j^* \geqslant \delta \sum_{1 \leqslant j < i \leqslant n} (a_i/\delta - 1) b_j x_i^* \oplus x_j^* \\ &= F(x^*) - \delta \sum_{1 \leqslant j < i \leqslant n} b_j x_i^* \oplus x_j^*. \end{aligned}$$

Thus

$$\begin{aligned} F(x^*) - F(y) &\leqslant \delta \sum_{1 \leqslant j < i \leqslant n} b_j x_i^* \oplus x_j^* \\ &= \varepsilon\overline{a}/2 \sum_{1 \leqslant j < i \leqslant n} b_j x_i^* \oplus x_j^* \leqslant \varepsilon\overline{a}/2 \sum_{1 \leqslant j \leqslant n-1} (n-j) b_j. \end{aligned}$$

By Lemma 2

$$F(x^*) \geqslant 1/2 \sum_{1 \leqslant j < i \leqslant n} a_i b_j.$$

Therefore, it suffices to prove that

$$\overline{a} \sum_{1 \leqslant j \leqslant n-1} (n-j) b_j \leqslant \sum_{1 \leqslant j < i \leqslant n} a_i b_j = \sum_{1 \leqslant j \leqslant n-1} b_j (a_{j+1} + \cdots + a_n).$$

This holds since

$$0 \leqslant a_2 \leqslant a_3 \leqslant \cdots \leqslant a_n$$

and consequently

$$\overline{a} \leqslant (a_{j+1} + \cdots + a_n)/(n-j). \qquad \square$$

4. Dominating and optimal solutions

If *both* a and b are either ascending or descending, then the matrix $(a_i b_j)$ is *anti-Monge*, i.e.

$$a_i b_j + a_k b_l \geqslant a_i b_l + a_k b_j$$

for $1 \leqslant i < k \leqslant n$ and $1 \leqslant j < l \leqslant n$. The two cases are NP-hard, the former since MAKS (and also CTV) are NP-hard, the latter since MAKS is NP-hard. However, in both cases ε-solutions can be computed in $O(n^2/\varepsilon)$ time by the fully polynomial time approximation schemes of Section 2. On the other hand *descending* a and *ascending* b result in $(a_i b_j)$ being *Monge*, i.e.

$$a_i b_j + a_k b_l \leqslant a_i b_l + a_k b_j$$

for $1 \leqslant i < k \leqslant n$ and $1 \leqslant j < l \leqslant n$. Though the schemes of Section 2 *fail* for half-products with ascending b and descending a, we show that development of such schemes is *superfluous* since the alternating solutions are optimal for any half-product $F(x)$ in this case. An *alternating* solution assigns one value to the even-indexed variables and the other to the odd-indexed variables. We have the following result.

Theorem 3. *The symmetric half-product* $F_{a,b}(x) = \sum_{1 \leqslant j < i \leqslant n} a_i b_j x_i \oplus x_j$ *is maximized by an alternating solution for any vectors* a *and* b *with* $a_1 \geqslant a_2 \geqslant \cdots \geqslant a_n \geqslant 0$ *and* $0 \leqslant b_1 \leqslant b_2 \leqslant \cdots \leqslant b_n$.

Proof. The proof proceeds by contradiction. Suppose that the theorem does not hold, then there exist an n and two vectors $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$ with $a_1 \geqslant a_2 \geqslant \cdots \geqslant a_n \geqslant 0$ and $0 \leqslant b_1 \leqslant b_2 \leqslant \cdots \leqslant b_n$ for which no optimal solution of $F_{a,b}(x)$ is alternating. Let us consider counterexamples with the smallest n, and take the one with minimum $A + B$ among them, where $A = \sum_{i=1}^{n} a_i$ and $B = \sum_{i=1}^{n} b_i$. Without loss of generality we may assume that $a_n > 0$ and $b_1 > 0$. We can rewrite $F_{a,b}(x)$ as follows:

$$F_{a,b}(x) = F_{a-e,b}(x) + F_{e,b}(x),$$

where both $a - e$ and e are descending and non-negative. Therefore, if $A > n$, then each half-products on the right hand side has a smaller than $A + B$ coefficient sum, and thus by the inductive hypothesis, both of them are minimized by the same alternating solution. Otherwise, that is if $A = n$, then let us rewrite $F_{a,b}(x)$ as follows:

$$F_{a,b}(x) = F_{a,b-e}(x) + F_{a,e}(x),$$

where both $b - e$ and e are ascending and non-negative. Therefore, if $B > n$, then each half-products on the right-hand side has a smaller than $A + B$ coefficient sum, and thus by the inductive hypothesis, both of them are minimized by the same alternating solution. Otherwise, that is if $B = n$, then both $a = e$ (since $A = n$) and $b = e$ and we have

$$F_{a,b}(x) = F_{e,e}(x) = \sum_{1 \leqslant j < i \leqslant n} x_i \oplus x_j$$

for which both alternating solutions are clearly optimal since both set exactly $\lceil n/2 \rceil \lfloor n/2 \rfloor$ terms on the right hand side to 1, which maximizes $F_{e,e}(x)$. This again leads to a contradiction and proves the theorem. □

Descending b and *ascending a* also result in $(a_i b_j)$ being *Monge*, however, the maximization problem for this case remains open. Consequently, it is not clear whether it is the Monge property that makes the ordered, symmetric half-products easy to maximize. However, we have the following partial results in Lemmas 3 and 4 and Theorem 4.

We begin by investigating ordered, symmetric half-products with either a or b being a unit vector $e = (1, \ldots, 1)$. Then, generally, optimal solutions can be found in $O(n^2)$ time by dynamic programming algorithms given in the proofs of Theorems 1 and 3. However, the descending b results in a set of dominating solutions which can be characterized more precisely. These dominating solutions are *block* solutions of the form $B_i = 0^i 1^{n-i}$, for $i = 1, \ldots, n-1$, where $0^i 1^{n-i}$ means $x_1 = \cdots = x_i = 0$ and $x_{i+1} = \cdots = x_n = 1$. Actually, solutions for descending b are dominated by blocks $B_1, \ldots, B_{\lfloor n/2 \rfloor}$.

Lemma 3. *At least one of the blocks* $B_1, \ldots, B_{\lfloor n/2 \rfloor}$ *maximizes* $f(x) = \sum_{1 \leqslant j < i \leqslant n} b_j x_i \oplus x_j$ *for b such that* $b_1 \geqslant b_2 \geqslant \cdots \geqslant b_n \geqslant 0$.

Proof. By the symmetry of $f(x)$ it is sufficient to consider only solutions x with $0 < n_0 \leqslant n_1$, where n_0 and n_1 are the number of 0's and 1's respectively in x. Thus, $n_0 = 1, \ldots, \lfloor n/2 \rfloor$. For a solution x and j, $j = 1, \ldots, n-1$ we have

$$\sum_{j < i \leqslant n} b_j x_i \oplus x_j \leqslant n_1 b_j.$$

Since there are exactly $n_0 n_1$ products $x_i \oplus x_j$ equal 1 in $f(x)$, then

$$\sum_{1 \leqslant j < i \leqslant n} b_j x_i \oplus x_j \leqslant n_1 (b_1 + \cdots + b_{n_0}).$$

However,

$$\sum_{1 \leqslant j < i \leqslant n} b_j y_i \oplus y_j = n_1 (b_1 + \cdots + b_{n_0}),$$

where $y = B_{n_0}$, which proves the lemma. □

On the other hand solutions for ascending a are dominated by blocks $B_{\lceil n/2 \rceil}, \ldots, B_{n-1}$ since a proof similar to this of Lemma 3 shows the following lemma.

Lemma 4. *At least one of the blocks* $B_{\lceil n/2 \rceil}, \ldots, B_{n-1}$ *maximizes* $g(x) = \sum_{1 \leqslant j < i \leqslant n} a_i x_i \oplus x_j$ *for a such that* $0 \leqslant a_1 \leqslant a_2 \leqslant \cdots \leqslant a_n$.

Finally, for general non-negative ascending a and descending b we have the following theorem.

Theorem 4. *The block $B_{n/2}$ maximizes $F(x)=\sum_{1\leqslant j<i\leqslant n} a_i b_j x_i \oplus x_j$ over all vectors x with equal numbers of 0's and 1's.*

Proof. For a solution $x \neq B_{n/2}$ with equal numbers of 0's and 1's, define an n by n matrix X as follows:

$$X_{ij}=\begin{cases} x_i \oplus x_j & \text{for } 1\leqslant j<i\leqslant n,\\ 0 & \text{otherwise.}\end{cases}$$

Also, define

$$H(X)=\sum_{1\leqslant j<i\leqslant n} a_i b_j X_{ij}=\sum_{1\leqslant j<i\leqslant n} a_i b_j x_i \oplus x_j.$$

We now show how to exchange the 1's in X to obtain a matrix Y such that

$$Y_{ij}=\begin{cases} y_i \oplus y_j & \text{for } 1\leqslant j<i\leqslant n,\\ 0 & \text{otherwise,}\end{cases}$$

where $y=B_{n/2}$, and

$$H(X)\leqslant H(Y).$$

Let k^* be the largest k such $X_{kj}=1$ for some $n/2<j<k\leqslant n$. Let $X_{k^*l}=0$ for some $l\leqslant n/2$. Consider the following two cases:

Case 1: $\sum_{1\leqslant i\leqslant n} X_{il}<n/2$.

Case 2: $\sum_{1\leqslant i\leqslant n} X_{il}=n/2$.

In Case 1, set $X_{k^*j}:=0$ and $X_{k^*l}:=1$. In Case 2, let $X_{i^*l}=1$ for some $i^*<n/2$. Set $X_{i^*l}:=0$ and $X_{k^*l}:=1$, and $X_{k^*j}:=0$ and $X_{i^*j}:=1$.

The value of H does not decrease as a result of the exchange since

in Case 1, $a_{k^*}b_l \geqslant a_{k^*}b_j$ for $l<j$, and

in Case 2

$$a_{k^*}b_l + a_{i^*}b_j \geqslant a_{i^*}b_l + a_{k^*}b_j$$

$$(a_{k^*}-a_{i^*})(b_l-b_j)\geqslant 0$$

for $l<j$ and $k^*>i^*$. By applying the exchange sufficiently many times we obtain a matrix X' with $X'_{ij}=0$ for $n/2<i,\ j\leqslant n$, and $\sum_{1\leqslant i\leqslant n} X'_{ij}\leqslant n/2$ for $1\leqslant j\leqslant n/2$. Because of the latter, we can move the 1's in a column $1\leqslant j\leqslant n/2$ "down" to rows $n/2+1,\ldots,n$. Thus, we may assume that $X'_{ij}=0$ for $1\leqslant i,\ \ j\leqslant n/2$, and

$$H(X)\leqslant H(X').$$

Finally, we observe that

$$a_{n/2+1}b_{n/2}\geqslant a_i b_j$$

for $1\leqslant i\leqslant n/2$ and $n/2+1\leqslant j\leqslant n$. This allows us to move all 1's from the square $1\leqslant i\leqslant n/2$, $n/2+1\leqslant j\leqslant n$ to the square $n/2+1\leqslant i\leqslant n$ and $1\leqslant j\leqslant n/2$ without decreasing the value of H. Consequently, we have all the 1's placed in the square $n/2+1\leqslant i\leqslant n$ and $1\leqslant j\leqslant n/2$

without decreasing the value of H. In other words we reach the block $B_{n/2}$, which proves the theorem. □

Though Theorem 4, and Lemmas 5 and 6 indicate that blocks make up a dominating set for special ordered symmetric half-products $F(x)$ with descending b and ascending a, this does not hold true generally as the following counterexample, Sivignon [26], shows: let $n = 5$, $a = (2, 2, 2, 2, 3)$ and $b = (3, 2, 2, 2, 2)$. Then, the best blocks B_2 and B_3 result in 35 whereas solution $x = (0, 1, 1, 1, 0)$ gives 36.

It is worth noticing that both fully polynomial time approximation schemes developed in Section 2 work for half-products with descending b and ascending a though the complexity of their minimization remains open.

5. Conclusions

We studied the problem of minimizing ordered, symmetric half-products. The problem is NP-hard in the ordinary sense. We presented fully polynomial time approximation schemes running in $O(n^2/\varepsilon)$ time for the problem with descending b *or* ascending a. On the other hand, ascending b *and* descending a result in the *alternating* solutions being optimal for the problem. However, the complexity of the problem with descending b *and* ascending a remains open. Though, some partial characterization of dominating solutions for this case has been shown. It is interesting to note that both ascending b *and* descending a and descending b *and* ascending a result in (a_i, b_j) being *Monge*. However, it remains open whether the Monge property is sufficient to grant polynomial solvability for *both* cases, or even whether the Monge property has anything to do with their complexity.

Acknowledgements

This research has been supported by the NSERC research Grant OGP0105675. The author would like to thank anonymous referees for their insightful comments that resulted in an improved paper.

Appendix

MAKS: Find a schedule of n jobs with processing times $p_1, \ldots, p_n$ on two identical machines to minimize makespan.

Equivalent half-product:

$$MAKS(x) = \sum_{1 \leqslant j < i \leqslant n} p_i p_j x_i \oplus x_j.$$

Complexity and approximation: NP-hard in the ordinar sense, Karp [15]. Fully polynomial time approximation schemes, Ibarra and Kim [12], Lawler [20], and Sahni [25].

WCT: Find a schedule of n jobs with processing times $p_1, \ldots, p_n$ and weights $w_1, \ldots, w_n$ on two identical machines to minimize weighted completion time $\sum_{i=1}^{n} w_i C_i$.

Equivalent half-product:

$$F(x) = \sum_{1 \leqslant j < i \leqslant n} w_i p_j x_i \oplus x_j,$$

where the jobs are ordered so that

$$\frac{w_1}{p_1} \geqslant \frac{w_2}{p_2} \geqslant \cdots \geqslant \frac{w_n}{p_n}.$$

Complexity and approximation: NP-hard in the ordinar sense, Bruno et al. [5]. Fully polynomial time approximation schemes, Sahni [25], and Section 2 of this paper.

WET: Find a schedule of n jobs with processing times $p_1, \ldots, p_n$ and weights $w_1, \ldots, w_n$ on a single machine to minimize total weighted earliness/tardiness, $\sum_{i=1}^{n} w_i |C_i - d|$, for a given common unrestrictive due date d.

Equivalent half-product:

$$G(x) = \sum_{1 \leqslant j < i \leqslant n} w_i p_j x_i \oplus x_j + \sum_{1 \leqslant i \leqslant n} w_i p_i x_i,$$

where the jobs are ordered so that

$$\frac{w_1}{p_1} \geqslant \frac{w_2}{p_2} \geqslant \cdots \geqslant \frac{w_n}{p_n}.$$

Complexity and approximation: NP-hard in the ordinar sense, Hall and Posner [9]. Fully polynomial time approximation scheme, Kovalyov and Kubiak [16].

CTV: Find a schedule of $n+1$ jobs with processing times $p_1, \ldots, p_n, p_{n+1}$ on a single machine to minimize completion time variance $1/(n+1)\sum_{i=1}^{n+1} (C_i - \overline{C})^2$, where $\overline{C}$ is the average completion time.

Equivalent half-product:

$$G(x) = \sum_{2 \leqslant j < i \leqslant n} \beta_i \gamma_j x_i \oplus x_j,$$

where $\beta_i = (n - i + 1)p_i + \sum_{j=1}^{i} p_j$ and $\gamma_j = j p_j - \sum_{k=1}^{j} p_k$ and the jobs are ordered so that $p_1 \leqslant \cdots \leqslant p_n \leqslant p_{n+1}$.

Complexity and approximation: NP-hard in the ordinar sense, Kubiak [17]. Fully polynomial time approximation schemes, Badics and Boros [3], Kubiak et al. [19].

AWCTV: Find a schedule of n jobs with processing times $p_1, \ldots, p_n$ and weights $w_1, \ldots, w_n$ on a single machine to minimize weighted variance of completion time $1/n$ $\sum_{i=1}^{n} w_i (C_i - \overline{C})^2$, where $\overline{C}$ is the weighted average completion time $\overline{C} = 1/W \sum_{j=1}^{n} w_j C_j$, and the jobs are agreeably weighted, that is their processing times and weights can be ordered as follows

$$p_1 \leqslant p_2 \leqslant \cdots \leqslant p_n \quad \text{and} \quad w_1 \geqslant w_2 \geqslant \cdots \geqslant w_n.$$

Equivalent half-product:

$$G(x) = \sum_{2 \leqslant j < i \leqslant n} \beta_i \gamma_j x_i \oplus x_j - \sum_{2 \leqslant i \leqslant n} e_i y_i,$$

where

$$\beta_i = \overline{W}_{i+1} p_i + w_i P_i, \quad \gamma_j = W_j p_j - w_j P_j,$$

$$e_i = W p_i (P_i w_i + \delta_{i+1}) - \delta_1 \beta_i, \quad \delta_i = \sum_{i \leqslant j \leqslant n} w_j p_j,$$

and $W_i = \sum_{1 \leqslant j \leqslant i} w_i$, $P_i = \sum_{1 \leqslant j \leqslant i} p_i$, for $i = 1, 2, \ldots, n$, $\overline{W}_i = \sum_{i \leqslant j \leqslant n} w_j$ for $i = 1, 2, \ldots, n+1$, and $W = \sum_{1 \leqslant j \leqslant n} w_j$.

Complexity and approximation: NP-hard in the ordinar sense, Kubiak [17]. Fully polynomial time approximation schemes, Cai [6], Woeginger [28], Cheng and Kubiak [7].

CONT: Find a schedule of n jobs on a single machine to minimize the sum of the total weighted completion time $\sum_{j=1}^{n} w_j C_j$, and the total weighted processing time compression $\sum_{j=1}^{n} v_j (u_j - p_j)$, $TWC = \sum_{j=1}^{n} w_j C_j + \sum_{j=1}^{n} v_j (u_j - p_j)$. The processing time of job j is a variable $p_j \in [0, u_j]$, $j = 1, \ldots, n$. The solution is to find both the optimal values of job processing times $p = (p_1, \ldots, p_n)$ and an optimal permutation of jobs π.

Equivalent half-product:

$$TWC(x) = \sum_{1 \leqslant j < i \leqslant n} w_i u_j x_i x_j + \sum_{j=1}^{n} v_j u_j \overline{x}_j,$$

where $u_1/w_1 \leqslant \cdots \leqslant u_n/w_n$.

Complexity and approximation: NP-hard in the ordinary sense, Janiak et al. [13]. Fully polynomial time approximation schemes Janiak et al. [13].

References

[1] D.J. Amit, Modeling Brain Function: The World of Attractor Neural Networks, Cambridge University Press, Cambridge, 1989.

[2] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy, Proof verification and hardness of approximation problems, Proceedings of the 33rd Annual Symposium on Foundations of Computer Science, 1992, pp. 14–23.

[3] T. Badics, E. Boros, Minimization of half-products, Math. Oper. Res. 23 (3) (1998) 649–660.

[4] E. Boros, P.L. Hammer, The max-cut problem and quadratic 0–1 optimization; polyhedral aspects, relaxations and bounds, Ann. Oper. Res. 33 (1991) 151–180.

[5] J. Bruno, E.G. Coffman Jr., R. Sethi, Scheduling independent tasks to reduce mean finishing time, Comm. ACM 17 (1974) 382–387.

[6] X. Cai, Minimization of agreeably weighted variance in single machine systems, Eur. J. Oper. Res. 85 (1995) 576–592.

[7] J. Cheng, W. Kubiak, A half-product based approximation scheme for agreeably weighted completion time variance, Eur. J. Oper. Res. 162 (2005) 45–54.

[8] W.L. Eastman, S. Even, I.M. Isaacs, Bounds for the optimal scheduling on n jobs on m machines, Management Sci. 11 (1964) 268–279.

[9] N.G. Hall, M.E. Posner, Earliness-tardiness scheduling problems I: weighted deviation of completion times about a common due date, Oper. Res. 39 (1991) 836–846.

[10] P.L. Hammer, P. Hansen, P.M. Pardalos, D.J. Rader Jr., Maximizing the product of two linear functions in 0–1 variables, Optimization 51 (2002) 511–537.

[11] P. Hansen, B. Jaumard, V. Mathon, Constrained nonlinear 0–1 programming, ORSA J. Comput. 5 (2) (1993) 97–119.

[12] O. Ibarra, C. Kim, Fast approximation algorithms for the knapsack and sum of subsets problems, J. Assoc. Comput. Mech. 22 (1975) 463–468.
[13] A. Janiak, M. Kovalyov, W. Kubiak, F. Werner, Positive half-products and scheduling with controllable processing times, Eur. J. Oper. Res. (2003), accepted for publication.
[14] B. Jurisch, W. Kubiak, J. Józefowska, Algorithms for minclique scheduling problems, Discrete Appl. Math. 72 (1997) 115–139.
[15] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, 1972, pp. 85–104.
[16] M.Y. Kovalyov, W. Kubiak, A fully polynomial approximation scheme for weighted earliness-tardiness problem, Oper. Res. 47 (1999) 757–761.
[17] W. Kubiak, Completion time variance minimization on a single machine is difficult, Oper. Res. Lett. 14 (1993) 49–59.
[18] W. Kubiak, New results on the completion time variance minimization, Discrete Appl. Math. 58 (1995) 157–168.
[19] W. Kubiak, J. Cheng, M.Y. Kovalyov, Fast fully approximation schemes for minimizing completion time variance, Eur. J. Oper. Res. 137 (2002) 303–309.
[20] E. Lawler, Fast approximation algorithms for knapsack problems, in: Proceedings of the 18th Annual Symposium on Foundations of Computer Science, Rhode Island, 1977, pp. 206–213.
[21] D.C. Mattis, Solvable spin systems with random interaction, Phys. Lett. 56A (1976) 412.
[22] G.L. Nemhauser, L.A. Wolsey, Integer and Combinatorial Optimization, Wiley, New York, 1988.
[23] G.L. Nemhauser, L.A. Wolsey, M.L. Fisher, An analysis of approximations for maximizing submodular set functions—I, Math. Programming 14 (1978) 265–294.
[24] I. Rosenberg, 0–1 Optimization and non-linear programming, Revue Française d'Automatique, Informatique et Recherche Operationnelle 6 (1972) 95–97.
[25] S. Sahni, Algorithms for scheduling independent tasks, J. Assoc. Comput. Mech. 23 (1976) 116–127.
[26] I. Sivignon, private communication, 2000.
[27] I. Wegener, The Complexity of Boolean Functions, Wiley, New York, 1987.
[28] G.J. Woeginger, An approximation scheme for minimizing agreeably weighted variance on a single machine, INFORMS J. Comput. 11 (1999) 211–216.
[29] G.J. Woeginger, When does a dynamic programming formulation guarantee the existence of an FPTAS?, INFORMS J. Comput. 12 (2000) 57–75.